

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 07-200542

(43)Date of publication of application : 04.08.1995

(51)Int.Cl.

G06F 17/16

G06F 9/38

(21)Application number : 05-338117

(71)Applicant : FUJITSU LTD

(22)Date of filing : 28.12.1993

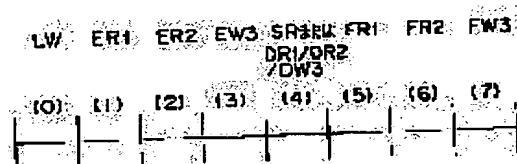
(72)Inventor : NAKATANI SHOJI
MOCHIYAMA TAKASHI
KURODA KOJI
KONNO KATSUHIKO
ATSUMI HIROAKI

(54) VECTOR PROCESSOR

(57)Abstract:

PURPOSE: To improve the performance of a vector processor which successively calculate a series of data stored in a main storage device after supplying them to an arithmetic pipeline by performing the arithmetic processing without deteriorating the overall computing efficiency for the arithmetic pipeline of small throughput and by separating the converging period of a vector macroinstruction from the outrun prevention control in regard of the sum total, the retrieval, etc.

CONSTITUTION: If a pipeline that has small throughput compared with other pipelines is detected, a bank management part prescribes the access timing to a vector register of the pipeline. Under such conditions, the pipeline of small throughput uses the timing (SR) set for a memory access pipeline as the access timing (DR1/DR2/DW3) set to the vector register. In addition, a changing part is added on a transmission line of an outrun prevention signal. Thus the outrun prevention signal is variable by a notification signal which notifies a converging period of the relevant instruction.



LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision
of rejection]

[Date of requesting appeal against examiner's
decision of rejection]

[Date of extinction of right]

Copyright (C); 1998,2003 Japan Patent Office

3/4/95

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平7-200542

(43) 公開日 平成7年(1995)8月4日

(51) Int.Cl.⁶

G 0 6 F 17/16
9/38

識別記号

3 1 0 J

庁内整理番号

F I

G 0 6 F 15/ 347

技術表示箇所

H

審査請求 未請求 請求項の数 5 O L (全 21 頁)

(21) 出願番号

特願平5-338117

(22) 出願日

平成5年(1993)12月28日

(71) 出願人

000005223

富士通株式会社

神奈川県川崎市中原区上小田中1015番地

(72) 発明者

中谷 彰二

神奈川県川崎市中原区上小田中1015番地

富士通株式会社内

(72) 発明者

持山 貴司

神奈川県川崎市中原区上小田中1015番地

富士通株式会社内

(72) 発明者

黒田 浩二

神奈川県川崎市中原区上小田中1015番地

富士通株式会社内

(74) 代理人

弁理士 真田 有

最終頁に続く

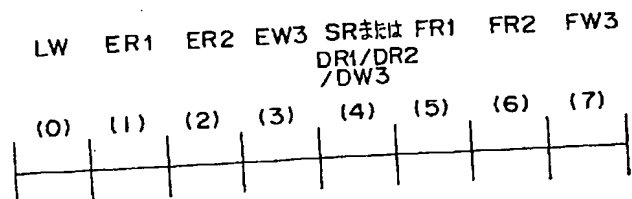
(54) 【発明の名称】 ベクトル処理装置

(57) 【要約】

【目的】 本発明は、主記憶装置上の一連のデータを順次演算パイプラインに入力して演算するベクトル処理装置に関し、スループットの少ない演算パイプラインについて全体の演算効率を落とすことなく演算処理可能として、演算スループットの向上をはかるとともに、総和、検索等のベクトルマクロ命令の収束期間を逾越し防止制御から切り離すようにして性能の向上をはかることを目的とする。

【構成】 他のパイプラインに比べ低スループットのパイプラインが1つでもある場合、バンク管理部によりパイプラインのベクトルレジスタへのアクセスタイミングを規定する際、低スループットのパイプラインがベクトルレジスタをアクセスするタイミング(DR1/DR2/DW3)として、メモリアクセスパイプライン用のタイミング(SR)を用いるように構成するほか、追越し防止信号の伝送路上に変更部を追加し、該当命令の収束期間の通知信号により信号を変更可能に構成する。

第1の発明の原理説明図



【特許請求の範囲】

【請求項1】 インターリーブされた複数のバンク単位に複数のエレメント・データを記憶するベクトル・レジスタ(1-0, ..., 1-n)と、

該ベクトル・レジスタ(1-0, ..., 1-n)の各エレメント・データをアクセスする複数の演算パイプライン(5-0, ..., 5-m)および1つまたは複数のメモリ・アクセス・パイプライン(2)と、

前記の演算パイプライン(5-0, ..., 5-m)およびメモリ・アクセス・パイプライン(2)が各バンク単位をアクセスできるタイミングを示すバンクスロットを管理するバンク管理部(7)とを有し、

前記の演算パイプライン(5-0, ..., 5-m)およびメモリ・アクセス・パイプライン(2)が該ベクトル・レジスタ(1-0, ..., 1-n)の各バンク単位を順次アクセスして各エレメント・データを処理するベクトル処理装置において、

前記複数の演算パイプライン(5-0, ..., 5-m)の中に、他の演算パイプラインに比べて演算スループットの低い演算パイプライン(15)を少なくとも1つ有し、

該バンク管理部(7)により、前記の演算パイプライン(5-0, ..., 5-m)およびメモリ・アクセス・パイプライン(2)が該ベクトル・レジスタ(1-0, ..., 1-n)をアクセスするための各アクセス・タイミングを規定する際に、前記複数の演算パイプライン(5-0, ..., 5-m)のうちの演算スループットの低い演算パイプライン(15)が該ベクトル・レジスタ(1-0, ..., 1-n)をアクセスするタイミングは、前記メモリ・アクセス・パイプライン(2)として割り付けられたタイミングを用いることを特徴とする、ベクトル処理装置。

【請求項2】 前記複数の演算パイプライン(5-0, ..., 5-m)のうちの演算スループットの低い演算パイプライン(15)が該ベクトル・レジスタ(1-0, ..., 1-n)をアクセスするタイミングは、前記メモリ・アクセス・パイプライン(2)のうちのストア・パイプライン(2B)が該ベクトル・レジスタ(1-0, ..., 1-n)から主記憶部(20)へストア動作する読み出しタイミングのバンクスロットに対して割り当てられることを特徴とする、請求項1記載のベクトル処理装置。

【請求項3】 インターリーブされた複数のバンク単位に複数のエレメント・データを記憶するベクトル・レジスタ(21)と、

該ベクトル・レジスタ(21)上のデータを入力オペランドとするか、もしくは、演算結果を該ベクトル・レジスタ(21)に書き込む1つまたは複数の演算パイプライン(22)と、

主記憶部(24)から該ベクトル・レジスタ(21)へデータを転送する1つまたは複数のロード・パイプライン(23)とを有し、

該ロード・パイプライン(23)から該ベクトル・レジスタ(21)へデータを転送する命令の実行中に、該ロード・パイプライン(23)が該ベクトル・レジスタ(21)に書き込んだデータを入力オペランドとする後続の演算命令を演算パイプライン(22)が実行する場合、命令の実行順序を保証するために、ロード・パイプライン(23)の実行を後続の演算パイプライン(22)の処理が追いつく条件を検出した時に、全ての演算パイプライン(22)の実行を一時中断する追越し防止制御部(25)をそなえたベクトル処理装置において、

該ベクトル・レジスタ(21)からのデータ供給を受けるリード処理期間と、該リード処理期間後に結果をまとめ上げる収束期間とを必要とするベクトル命令について、該当ベクトル命令の収束処理を実行中の演算パイプライン(22)に対する追越し防止制御を行なわないように、該追越し防止制御部(25)から出力される追越し防止制御信号を変更する変更部(26)がそなえられていることを特徴とする、ベクトル処理装置。

【請求項4】 該変更部(26)が、該演算パイプライン(25)にそなえられていることを特徴とする、請求項3記載のベクトル処理装置。

【請求項5】 該当ベクトル命令の収束処理を実行中の演算パイプライン(22)が、基本演算器と収束を処理する付加演算器とを有する構成のもので、収束処理を該付加演算器により実行し、収束処理中、該基本演算器により後続の他の演算命令を実行できるものである場合には、収束処理中、該変更部(26)が、該付加演算器に対してのみ追越し防止制御を行なわないように、該追越し防止制御信号を変更することを特徴とする、請求項3または4に記載のベクトル処理装置。

【発明の詳細な説明】

【0001】 (目次)

産業上の利用分野

従来の技術(図19)

発明が解決しようとする課題(図19)

課題を解決するための手段(図1, 図2)

作用

実施例

(a) 第1実施例の説明(図3~図7)

(b) 第2実施例の説明(図8~図18)

発明の効果

【0002】

【産業上の利用分野】 本発明は、主記憶装置に格納されている一連のデータを順次演算パイプラインに入力して演算するベクトル処理装置に関する。

【0003】

【従来の技術】 一般に、ベクトル処理装置においては、例えば、ベクトルBに属するエレメント・データ b_0, b_1, \dots および/またはベクトルCに属するエレメント・データ c_0, c_1, \dots とを有し、

ータ c_0, c_1, \dots をパイプライン処理によって演算し、その結果得られた a_0, a_1, \dots をベクトルAに属するエレメント・データとして抽出することが行なわれる。この場合、主記憶装置から置換ロードして演算パイプラインに入力したり、演算パイプラインから主記憶装置に置換ストアしたりすることは、主記憶装置のアクセス速度に制限され、処理速度が遅くなってしまう。

【0004】このため、通常、主記憶装置と演算パイプラインとの間に、複数のバンク単位により構成されインターリーブ構造をもつベクトル・レジスタをそなえることが行なわれている。このベクトル・レジスタは、1つのベクトル・レジスタに属するエレメント・データの例えば第 i 番目のデータと第 $(i+1)$ 番目のデータとが互いに異なるバンク単位に格納されるように構成され、各バンク単位の読出出力が互いに異なるバスを介して演算パイプラインに供給されるように構成されるとともに、演算パイプラインから得られた演算結果を互いに異なるバスを介して各バンク単位に書き込むように構成されている。

【0005】この演算パイプラインが複数ある場合、演算パイプラインからの要請によってベクトル・レジスタへ同時に並列的にアクセスすることが可能な数は、例えば8バンク単位にインターリーブされているとすれば、8個まで可能である。従って、各演算パイプラインからベクトル・レジスタへのアクセスにおいて、同時に同一のバンク単位にアクセスしないように、また、各バンク単位ないし各演算パイプラインが効率よく動作するように、ベクトル・レジスタへのアクセス・タイミングを管理することが極めて重要となる。

【0006】ベクトル・レジスタへの各エレメント・データの格納に際し、例えば上述のエレメント・データ $b_0, b_1, \dots, c_0, c_1, \dots$ については同じエレメント番号のデータが演算される関係から、同じタイミングで読出を行なうと都合であるため、可能な限り異なったバンク単位に位置するように格納されている。従って、ベクトル・レジスタへアクセスするためのハードウェアでは、各エレメント・データの最初の格納アドレス（例えば第0番目のエレメントに対するバンク情報）について記憶する手段が必要であるとともに、同時に同一のバンク単位へアクセスしないような作用が必要となり、アクセス制御のためのハードウェアが複雑になってしまう。

【0007】そこで、従来より、インターリーブされた複数のバンク単位に複数のエレメント・データを記憶するベクトル・レジスタと、このベクトルレジスタの各エレメント・データをアクセスする演算パイプラインおよびメモリ・アクセス・パイプラインと、これらのパイプラインが各バンク単位をアクセスできるタイミングを示すバンクスロットを管理するバンク管理部とを有するベクトル処理装置において、各パイプラインを起動する時点で、バンク管理部からバンクスロット信号を送出する

ことにより、各パイプラインのアクセスタイミングを規定することが行なわれている（例えば特開昭57-31079号公報参照）。

【0008】このような従来のベクトル処理装置におけるバンクスロットのタイミング例について、図19を参照しながら説明する。この図19において、 $B_0 \sim B_7$ はそれぞれ8個のバンク単位を示し、 LW は、メモリ・アクセス・パイプラインのうちのロード・パイプラインが主記憶装置（メモリ）からベクトル・レジスタにデータを書き込むタイミングを示し、 SR は、メモリ・アクセス・パイプラインのうちのストア・パイプラインがベクトル・レジスタから主記憶装置へ読み出すタイミングを示す。

【0009】また、 E, F はそれぞれ例えば2種類の演算パイプライン（乗算もしくは加算パイプライン）が動作するバンクスロットの区別を示し、各 E, F に付された R, W はそれぞれ各オペランドによるベクトル・レジスタから各演算パイプラインへの読出（READ）、各演算パイプラインからベクトル・レジスタへの書込（WRITE）に対応するものであり、 R, W に付された数字1, 2, 3はそれぞれ各オペランド OP_1, OP_2, OP_3 の番号に対応している（ $OP_1 * OP_2 \Rightarrow OP_3$ ；*は演算子）。

【0010】さらに、 $(0) \sim (7)$ はカウント値を示している。この図19に示すように、従来、演算パイプラインのように固定パイプライン長として割り当てるバンクスロットには、 ER_1, ER_2, EW_3 が割り当てられ、メモリ・アクセス・パイプラインのようにパイプライン長が不定のパイプラインには、 LW または SR が割り当てられている。

【0011】一方、ベクトル処理装置では、処理を高速化するために、1つの命令の実行完了を待たずに後続の命令の実行を開始するが、処理スループットの異なるパイプライン間で結果オペランドと入力オペランドとに依存関係がある場合には、追越し防止制御を行なう必要がある。複数のパイプライン間のデータリンクの状況を正確に確認して適切な追越し防止制御を行なうには、物量的なインパクトが大きく、遅延時間も大きくなるので、現実的ではない。

【0012】そのため、適切さは欠くが簡便な手法が従来より使用されている。その1つが、少なくとも1つの演算パイプラインがリンク状態にあるときには、追越し禁止制御を行なう際に、対象とする演算パイプラインのみでなく、全演算パイプラインに対して追越し禁止制御を適用するという手法である。つまり、従来の追越し制御防止部は、ロード命令が他のパイプラインとリンク動作（チェイニングともいう）を開始するという情報を命令発信／管理部から受け取ると、そのロード命令が完了するまで、追越し防止制御を活性化する。追越し防止制御部は、リンク動作しているロード・パイプラインが、

メモリ・バス・コンフリクト等の要因で、所定のスループットを下回るおそれがあることを検出すると、追越し防止制御信号を立ち上げることににより、演算パイプライン、ストア・パイプラインの実行を中断させる。

【0013】

【発明が解決しようとする課題】ところで、図19に示すようにバンクスロットのタイミングを設定する場合、固定長パイプラインの中でも、乗算パイプラインや加算パイプラインは、各サイクル、演算結果を得ることが可能であるが、割算パイプラインのように各サイクル毎に数ビットの結果しか得られないものもある（乗算／加算時に比べ、1／3以下のパイプラインで、例えば1／7のスループット）。

【0014】このようなスループットの少ない演算パイプライン、つまり読出あるいは書込に要するバンクの使用時間が少ない演算パイプラインが、バンクスロット（ER1, ER2, EW3, FR1, FR2, FW3）を占有することは、全体の演算効率を落とし、ベクトル処理装置全体のスループットの低下を招くという課題があった。

【0015】また、従来の追越し防止制御方式では、他のパイプラインの動作フェイズには関与しないため、過剰にパイプラインを止めてしまうことがある。例えば、総和命令、検索命令、抽出命令のように最終的に1つの結果を求める演算では、演算途中で発生する中間的な結果をまとめ上げるための収束期間が存在するが、このような収束期間は、原理的に前述のリンク動作とは独立に動作することが可能であるにもかかわらず、従来の追越し防止制御では、その追越し防止制御信号による処理停止のために収束処理までも停止してしまい、処理速度の低下を招くなどの課題もあった。

【0016】本発明はこのような課題に鑑み創案されたもので、本発明の第1の目的は、スループットの少ない演算パイプラインについて、全体の演算効率を落とすことなく演算処理を行なえるようにして、演算スループットの向上をはかったベクトル処理装置を提供することである。また、本発明の第2の目的は、演算パイプラインが総和、検索等のベクトルマクロ命令の収束処理のシーケンス実行中である場合には、追越し防止制御を行なわないように制御することにより、処理速度の改善をはかったベクトル処理装置を提供することである。

【0017】

【課題を解決するための手段】図1は第1の発明の原理説明図である。第1の発明のベクトル処理装置も、基本的には従来のベクトル処理装置と同様に、インターリーブされた複数のバンク単位に複数のエレメント・データを記憶するベクトル・レジスタと、このベクトル・レジスタの各エレメント・データをアクセスする複数の演算パイプラインおよび1つまたは複数のメモリ・アクセス・パイプラインと、これらのパイプラインが各バンク単

位をアクセスできるタイミングを示すバンクスロットを管理するバンク管理部とから構成され、演算パイプラインおよびメモリ・アクセス・パイプラインがベクトル・レジスタの各バンク単位を順次アクセスして各エレメント・データが処理されるようになっている。

【0018】そして、第1の発明では、複数の演算パイプラインの中に、他の演算パイプラインに比べて演算スループットの低い演算パイプラインを少なくとも1つ有する場合、図1に示すように、バンク管理部により、複数の演算パイプラインのうちの演算スループットの低い演算パイプラインがベクトル・レジスタをアクセスするタイミング（DR1, DR2, DW3）は、メモリ・アクセス・パイプラインとして割り付けられたタイミング、特にメモリ・アクセス・パイプラインのうちのストア・パイプラインがベクトル・レジスタから主記憶部へストア動作する読み出しタイミング（SR）のバンクスロットに対して割り当てられている（請求項1, 2）。なお、図1中の各符号は、図19により前述したものと同様であるが、Dは、演算スループットの低い演算パイプラインが動作するバンクスロットの区別を示している。

【0019】また、図2は第2の発明の原理ブロック図で、この図2において、21はインターリーブされた複数のバンク単位に複数のエレメント・データを記憶するベクトル・レジスタ、22はベクトル・レジスタ21上のデータを入力オペランドとするかもしくは演算結果をベクトル・レジスタ21に書き込む1つまたは複数の演算パイプライン、23は主記憶部24からベクトル・レジスタ21へデータを転送する1つまたは複数のロード・パイプラインである。

【0020】また、25は追越し防止制御部で、この追越し防止制御部25は、ロード・パイプライン23からベクトル・レジスタ21へデータを転送する命令の実行中に、ロード・パイプライン23がベクトル・レジスタ21に書き込んだデータを入力オペランドとする後続の演算命令を演算パイプライン22が実行する場合、命令の実行順序を保証するために、ロード・パイプライン23の実行を後続の演算パイプライン22の処理が追いつく条件を検出した時に、全ての演算パイプライン22の実行を一時中断するものである。

【0021】そして、第2の発明では、変更部26が新たにそなえられている。この変更部26は、ベクトル・レジスタ21からのデータ供給を受けるリード処理期間と、リード処理期間後に結果をまとめ上げる収束期間とを必要とするベクトル命令については、該当ベクトル命令の収束処理を実行中の演算パイプライン22に対する追越し防止制御を行なわないように、追越し防止制御部25から出力される追越し防止制御信号を変更するものである（請求項3）。

【0022】なお、この変更部26は、演算パイプライン

ン 22 にそなえてもよい (請求項 4)。また、該当ベクトル命令の収束処理を実行中の演算パイプライン 22 が、基本演算器と収束処理する付加演算器とを有する構成のもので、収束処理を該付加演算器により実行し、収束処理中、基本演算器により後続の他の演算命令を実行できるものである場合には、収束処理中、変更部 26 が、付加演算器に対してのみ追越し防止制御を行なわないように、追越し防止制御信号を変更してもよい (請求項 5)。

【0023】

【作用】 上述した第 1 の発明のベクトル処理装置 (請求項 1, 2) では、演算パイプラインのように固定長のパイプラインにおいても、特にスループットの少ないパイプラインを、メモリ・アクセス・パイプラインのような 1 つのバンクスロットしか使用しないパイプラインと共用することにより、演算パイプラインをオーバーラップさせて実行している。

【0024】 また、上述した第 2 の発明のベクトル処理装置 (請求項 3) では、追越し防止制御部 25 から追越し防止制御信号が出力された際に、演算パイプライン 22 が収束処理のシーケンスを実行中で、収束期間条件が成立している間は、変更部 26 により追越し防止制御部 25 からの追越し防止制御信号が変更され、収束処理中の演算パイプライン 22 に対する追越し防止制御が禁止される。

【0025】 なお、この変更部 26 を演算パイプライン 22 にそなえた場合には、収束処理を実行中の演算パイプラインが、追越し防止制御部 25 からの追越し防止制御のための信号を無視する形で、演算パイプライン 22 に対する追越し防止制御部 25 による追越し防止制御が禁止される (請求項 4)。また、収束処理中の演算パイプライン 22 が基本演算器と収束処理用の付加演算器とをもつものである場合には、変更部 26 により追越し防止制御部 25 からの追越し防止制御信号を変更することで、付加演算器に対してのみ追越し防止制御が禁止される (請求項 5)。

【0026】

【実施例】 以下、図面を参照して本発明の実施例を説明する。

(a) 第 1 実施例の説明

【図 3】 は本発明の第 1 実施例としてのベクトル処理装置を示すブロック図で、この図 3 において、 $1-0, 1-1, \dots, 1-n$ はそれぞれベクトル・レジスタで、各ベクトル・レジスタ $1-0, 1-1, \dots, 1-n$ は、それぞれ、インターリーブされたバンク単位 B_0, B_1, \dots, B_7 (本実施例では 8 バンク単位の場合を示す) に複数のエレメント・データを記憶するものである。

【0027】 2 は主記憶部 20 とベクトル・レジスタ $1-0, 1-1, \dots, 1-n$ との間において各エレメント・データを高速にロードないしストアすべくパイプライン

ン構成されたメモリ・アクセス・パイプラインで、ロード・パイプライン 2A およびストア・パイプライン 2B を有している。ここで、ロード・パイプライン 2A は、主記憶部 20 からのエレメント・データをベクトル・レジスタ $1-0, 1-1, \dots, 1-n$ へロードするためのものであり、ストア・パイプライン 2B は、ベクトル・レジスタ $1-0, 1-1, \dots, 1-n$ に格納されたエレメント・データを主記憶部 20 へストアするためのものである。

10 【0028】 3A はメモリ・アクセス・パイプライン 2 のロード・パイプライン 2A からのエレメント・データをベクトル・レジスタ $1-0, 1-1, \dots, 1-n$ における各バンク単位 $B_0 \sim B_7$ に書き込むための書込レジスタ、 $3B-0, 3B-1, \dots, 3B-m$ はそれぞれ後述する演算パイプライン $5-0, 5-1, \dots, 5-m$ からのエレメント・データ (演算結果) をベクトル・レジスタ $1-0, 1-1, \dots, 1-n$ における各バンク単位 $B_0 \sim B_7$ に書き込むための書込レジスタである。

20 【0029】 4A はベクトル・レジスタ $1-0, 1-1, \dots, 1-n$ に格納されたエレメント・データをメモリ・アクセス・パイプライン 2 のストア・パイプライン 2B へ読み出すための読出レジスタである。また、 $4B-0, 4C-0; 4B-1, 4C-1; \dots; 4B-m, 4C-m$ はそれぞれ後述する演算パイプライン $5-0, 5-1, \dots, 5-m$ 毎にそなえられた一対の読出レジスタで、各対の読出レジスタ $4B-0, 4C-0; 4B-1, 4C-1; \dots; 4B-m, 4C-m$ は、それぞれ、演算対象となる一対のエレメント・データを演算パイプライン $5-0, 5-1, \dots, 5-m$ に入力すべく、その一対のエレメント・データをベクトル・レジスタ $1-0, 1-1, \dots, 1-n$ における各バンク単位 $B_0 \sim B_7$ から読み出すためのものである。

【0030】 $5-0, 5-1, \dots, 5-m$ は演算パイプラインで、これらの演算パイプライン $5-0, 5-1, \dots, 5-m$ は、それぞれ、一対の読出レジスタ $4B-0, 4C-0; 4B-1, 4C-1; \dots; 4B-m, 4C-m$ を介して入力されたエレメント・データに対して四則演算等の演算処理を施し、その演算結果 (エレメント・データ) を出力するものである。

40 【0031】 そして、6 は各種のベクトル演算命令を出力する命令制御部、7 は命令制御部 6 からの命令を受けて動作するバンク管理部で、このバンク管理部 7 は、メモリ・アクセス・パイプライン 2 および演算パイプライン $5-0, 5-1, \dots, 5-m$ がベクトル・レジスタ $1-0, 1-1, \dots, 1-n$ における各バンク単位 $B_0 \sim B_7$ をアクセスできるタイミングを示すバンクスロットを管理するもので、各バンク単位 $B_0 \sim B_7$ へのアクセス・タイミングを規制するバンクスロット・カウンタ 7a を有している。

【0032】 次に、上述のようなベクトル処理装置の一

般的な動作について説明する。図 3 に示す本実施例のベクトル処理装置では、各ベクトル・レジスタ 1-0, 1-1, ..., 1-n は、各バンク単位 B 0 ~ B 7 にそれぞれ分散するように対応付けられている。そして、各ベクトル・レジスタ 1-0, 1-1, ..., 1-n に格納されるエレメント・データは、第 0 番目のデータがバンク単位 B 0 に記憶され、第 1 番目のデータがバンク単位 B 1 に記憶され、第 7 番目のデータがバンク単位 B 7 に記憶されるというように各バンク単位に順次記憶され、いわゆるインターリーブした形に格納され、同じナンバのデータが同じバンク単位に位置するように格納される。

【0033】例えば、ベクトル B に属するエレメント・データ b_0, b_1, \dots が主記憶部 20 からロードされてベクトル・レジスタ 1-1 内に格納されているものとし、またベクトル C に属するエレメント・データ c_0, c_1, \dots が同様にベクトル・レジスタ 1-2 に格納されているものとする。この状態で、例えば、命令制御部 6 からバンク管理部 7 に対して、ベクトル加算命令「OP 1 [#1 VR(i)] + OP 2 [#2 VR(i)] ⇒ OP 3 [#0 VR(i)]」が与えられたとすると、バンク管理部 7 により、次のごとく①~⑥の処理(図 4 参照)が実行される。なお、本実施例では、演算パイプライン(加算パイプライン) 5-0 が 3 段のステップ段数をもつものとする(図 4 参照)。

【0034】ここで、[#0 VR(i)], [#1 VR(i)], [#2 VR(i)] はそれぞれベクトル・レジスタ 1-0, 1-1, 1-2 の各バンク単位 B i に格納されるデータを意味し、上記ベクトル加算命令は、ベクトル・レジスタ 1-1 に格納された各エレメント・データと、ベクトル・レジスタ 1-2 に格納された各エレメント・データとを加算し、その加算結果(エレメント・データ)をベクトル・レジスタ 1-0 に格納する命令となっている。

【0035】①タイミング・サイクル(バンクスロット・カウンタ 7 a によりカウントされるカウント値に対応するもの) T 0, T 1, ... において、バンク単位 B 0, B 1, ..., B 7, ... に対して、順次、リード・アクセスが行なわれ、その結果、読出レジスタ 4 B-0, 4 C-0 を介して、エレメント・データ b_0, b_1, \dots および c_0, c_1, \dots が、順次、ベクトル・レジスタ 1-1, 1-2 から読み出される。

【0036】②タイミング・サイクル T 2 において、データ b_0 と c_0 とは演算パイプライン 5-0 のステップ I に入力される。

③タイミング・サイクル T 3 において、データ b_0 と c_0 とは演算パイプライン 5-0 のステップ II に入力されると同時に、データ b_1 と c_1 とが演算パイプライン 5-0 のステップ I に入力される。

【0037】④タイミング・サイクル T 4 において、データ b_0 と c_0 とは演算パイプライン 5-0 のステップ

III に入力され、データ b_1 と c_1 とは演算パイプライン 5-0 のステップ II に入力されると同時に、データ b_2 と c_2 とが演算パイプライン 5-0 のステップ I に入力される。

⑤タイミング・サイクル T 5 において、データ b_0 と c_0 との加算結果であるデータ a_0 が書込レジスタ 3 B-0 にセットされる。

【0038】⑥タイミング・サイクル T 6 において、このデータ a_0 が、ベクトル・レジスタ 1-0 のバンク単位 B 0 に書き込まれる。以下、順次得られるデータ a_1, a_2, \dots が書込レジスタ 3 B-0 にセットされ、書込レジスタ 3 B-0 にセットされたデータ a_1, a_2, \dots は、それぞれ、ベクトル・レジスタ 1-0 のバンク単位 B 1, B 2, ..., B 7, B 0, ... に順次書き込まれる。

【0039】ここで、演算パイプライン 5-0 では、同じナンバ(添字番号)のエレメント・データがステップ I に入力されるように、ベクトル B, C に属するエレメント・データの入力側にタイミングを合わせるためのバッファ・レジスタが 1 段設けられている。このように構成することによって、ベクトル B, C に関して加算し、その結果得られたベクトル A をバンク単位 B 0, B 1, ... の順にアクセスすることが可能になる。

【0040】ついで、前記の各バンク単位 B 0 ~ B 7 へのアクセス制御を簡略化すべく、そのアクセス制御を行なうバンク管理部 7 について、図 5, 図 6 を参照しながら説明する。図 5 において、11-1, 11-2, 11-3 はメモリ・アクセス・パイプライン 2 もしくは演算パイプライン 5-0, 5-1, ..., 5-m がベクトル・レジスタ 1-0, 1-1, ..., 1-n にアクセスするタイミング・サイクル(以下、バンクスロットという)を記憶する管理レジスタ、12 はバンクスロット割当回路、13 はバンクスロットを記憶しメモリ・アクセス・パイプライン 2 へ通知する通知レジスタ、14 は起動信号制御部である。

【0041】バンク管理部 7 では、各バンク単位 B 0 ~ B 7 にアクセスするパイプライン装置(メモリ・アクセス・パイプライン 2 もしくは演算パイプライン 5-0, 5-1, ..., 5-m)が同一バンク単位へ同時にアクセスすることのないように、さらには無駄な空き時間を生じない効率のよいアクセスが可能になるアクセス制御を実現するために、バンクスロット・カウンタ 7 a (1 回設けられている)を、常時カウントすることによって、バンクスロットと呼ばれるタイミング・サイクル T 0 ~ T 7 を規定し、そのカウンタ出力信号を各パイプライン装置へ通知している。

【0042】なお、このとき、バンク B 1 をアクセスするタイミングは、バンク B 0 よりも 1 サイクル遅れた状態であるため、本カウンタ 7 a によるカウント値が“1”の時、1 サイクル前でバンク B 0 をアクセスしていたパイプラインがバンク B 1 をアクセスする。従っ

て、バンクスロット・カウンタ 7 a によるカウント値は、バンク B 0 にアクセスするパイプラインの順を示している。

【0043】各管理レジスタ 11-1, 11-2, 11-3 は、各パイプライン装置が有する各バンク単位にデータ転送するためのチャンネル（アクセス要求）に対して、起動する時点において割り当てられるバンクスロット番号（B 1）を記憶する例えば 3 ビットの記憶素子（実際には記憶内容の有効／無効表示のためにさらに 1 ビットが必要）で構成されるものであり、パイプライン装置がベクトル・レジスタ 1-0, 1-1, ..., 1-n にアクセスしている期間、そのチャンネル（アクセス要求）に割り当てられたバンクスロット番号を記憶している。

【0044】バンクスロット割当回路 12 は、管理レジスタ 11-1, 11-2, 11-3 の出力と、バンクスロット・カウンタ 7 a の出力とによって、使用中のバンクスロット番号と現在のバンクスロット番号とを知り、起動のあるパイプライン装置に対して空き時間の最少となるようなバンクスロット番号を割り当てる選択回路である。

【0045】ここで、命令制御部 6 が、バンク管理部 7 に起動信号を与え、例えば、メモリ・アクセス・パイプライン 2 が主記憶部 20 へアクセスして読出データをベクトル・レジスタ 1-0 へデータ転送するように要求したとする。このとき、バンクスロット割当回路 12 は、メモリ・アクセス・パイプライン 2 を起動する時点において、当該チャンネル（アクセス要求）に相当する管理レジスタ 11-1, 11-2, 11-3 に、その選択したバンクスロット番号をセット番号として伝え記憶させる。

【0046】図 6 は上述の動作を詳細に説明するためのもので、この図 6 に示すように、メモリ・アクセス・パイプライン 2 が起動信号によって起動されると、エレメント・データ a_0, a_1, \dots, a_n 。（主記憶部 20 に格納されているものとする）にメモリ・アクセスが開始され、アクセス・タイム t_a の後に、メモリ・アクセス・パイプライン 2 のバッファ・レジスタ（図 5 には図示せず）にその読出内容がロードされる。

【0047】各パイプライン装置には、予め決められたバンクスロット（図 1, 図 7, 図 19 参照）が割り当てられているので、各パイプライン装置でバンクスロット・カウンタ 7 a の出力を参照し、所望のタイミングになった時点で、バッファ・レジスタに一時記憶していたエレメント・データ a_0, a_1, \dots, a_n をベクトル・レジスタ 1-0, ... へ順次転送する。最後のデータを転送すると、パイプライン終了信号によってリセット信号をバンク管理部 7 へ送出し、当該転送チャンネルに相当する管理レジスタ 11-1 または 11-2, 11-3 の内容をリセットし無効にする。

【0048】なお、バッファ・レジスタは、主記憶部 20 の読出出力をストローブするタイミングと、バンクスロットによりベクトル・レジスタ 1-0, ... に書き込むまでの期間を調整する複数のレジスタである。また、上述した例では、主記憶部 20 が他のアクセス装置からの要求によってビジー状態等であることを考慮すると、起動してからベクトル・レジスタ 1-0 を使用するまでの時間が一定でない。即ち、上述の例では、パイプライン長が不定な装置とすることができ、演算パイプライン 5-0, 5-1, ..., 5-m は、ステップ数（即ちパイプライン長）が固定であり、主記憶部 20 の状況に影響されず起動されてからベクトル・レジスタ 1-0 を使用するまでの時間は一定となる。この場合は、図 4 において説明した通り、演算パイプライン 5-0, 5-1, ..., 5-m が、ベクトル・レジスタ 1-0 にアクセスするタイミング関係は固定となっている。

【0049】このため、パイプライン長が固定の装置において、各チャンネル（アクセス要求）間のアクセス・タイミングのずれを認識することにより、図 5 に示すごとく、起動信号制御部 14 において、使用中のバンクスロット番号、バンクスロット・カウンタ 7 a の内容および起動信号に基づき、起動信号のタイミングでバンクスロットのタイミングを判断することが可能である。

【0050】さて、本実施例では、上述のような一般的な動作を行なうベクトル処理装置において、複数の演算パイプライン 5-0, 5-1, ..., 5-m の中に、他の演算パイプラインに比べて演算スループットの低い演算パイプラインとして、割算パイプライン 15（6 個の割算器 5 a ~ 5 f を有してなるパイプライン）を有する場合に、バンク管理部 7 は、図 7 に示すように、バンクスロットを管理している。

【0051】つまり、バンク管理部 7 は、各割算パイプライン 5 a ~ 5 f がベクトル・レジスタ 1-0, 1-1, ..., 1-n をアクセスするタイミング（DR 1, DR 2, DW 3）を、メモリ・アクセス・パイプライン 2 のうちのストア・パイプライン 2 B がベクトル・レジスタ 1-0, 1-1, ..., 1-n から主記憶部 20 へストア動作する読み出しタイミング（SR）のバンクスロットに対して割り当てている。

【0052】なお、図 7 において、各符号は、図 19 により前述したものと同様であるが、DR 1, DR 2 は割算パイプライン 15 における読出オペランドのタイミングを示し、DW 3 は割算パイプライン 15 における書込オペランドのタイミングを示している。割算パイプライン 15 が動作している間は、タイミングを時分割で使用する。また、割算パイプライン 15 とストア・パイプライン 2 B とはいずれか一方のみ動作するように制御する。さらに、E, F は演算パイプライン 5-0, 5-1, ..., 5-m のうちの加算パイプラインもしくは乗算パイプラインが動作するバンクスロットの割付タイミン

グを示す。

【0053】本実施例では、図7に示すように、2種類の演算パイプラインが動作している際に、各バンク単位B0~B7で割算パイプライン15の各割算器5a~5fは16タイミング・サイクル毎に到来する、ストア・パイプライン2Bがベクトル・レジスタ1-0, 1-1, ..., 1-nから主記憶部20へストア動作する読み出しタイミングSRのバンクスロットを共用して、割算パイプライン15による演算処理を実行している。

【0054】例えば、割算パイプライン15内の割算器5aでは、バンク単位B0の最初のストア読出タイミングSR（タイミング・サイクルT4）と、バンク単位B1の最初ストア読出タイミングSR（タイミング・サイクルT5）とで、ベクトル・レジスタ1-0, 1-1, ..., 1-nからのデータの読出（DR1, DR2）を行なう。そして、16タイミング・サイクル後のバンク単位B1の2周期目のストア読出タイミングSR（タイミング・サイクルT5）で、割算結果のベクトル・レジスタ1-0, 1-1, ..., 1-nへの書込（DW3）を行なった後、再び、バンク単位B2の2周期目のストア読出タイミングSR（タイミング・サイクルT6）と、バンク単位B3の2周期目のストア読出タイミングSR（タイミング・サイクルT7）とで、ベクトル・レジスタ1-0, 1-1, ..., 1-nからのデータの読出（DR1, DR2）を行ない、以下、同様の処理を繰り返す。

【0055】また、割算パイプライン15内の割算器5bでも、同様に、バンク単位B3の最初のストア読出タイミングSR（タイミング・サイクルT7）と、バンク単位B4の最初ストア読出タイミングSR（タイミング・サイクルT0）とで、ベクトル・レジスタ1-0, 1-1, ..., 1-nからのデータの読出（DR1, DR2）を行ない、16タイミング・サイクル後のバンク単位B4の2周期目のストア読出タイミングSR（タイミング・サイクルT0）で、割算結果のベクトル・レジスタ1-0, 1-1, ..., 1-nへの書込（DW3）を行ない、以下、同様の処理を繰り返す。その他の割算器5c, 5d, 5e, 5fについても同様に、タイミングの割付を行なって、割算処理が行なわれる。

【0056】このように、本発明の第1実施例のベクトル処理装置によれば、演算パイプラインのように固定長のパイプラインにおいても、特にスループットの少ないパイプライン、例えば割算パイプライン15を、メモリ・アクセス・パイプライン2のような1つのバンクスロットしか使用しないパイプラインと共用することにより、演算パイプラインをオーバーラップさせて実行し、全体の演算効率を落とすことなく演算処理を行なえ、演算スループットが大幅に向上することになる。

【0057】また、特に、メモリ・アクセス・パイプライン2の中でも、主記憶部20から読出を行なうロード

・パイプライン2Aは演算のソースとなるため、結果オペランドを格納するためのストア・パイプライン2Bのバンクスロットと共用することにより、ベクトル処理装置全体のスループットを向上させることができるのである。

【0058】なお、上述の実施例では、スループットの少ないパイプラインが割算パイプラインである場合について説明したが、本発明は、これに限定されるものではなく、スループットの少ないパイプラインがスクエア・ルート（2乗根）演算パイプライン等である場合にも同様に適用され、上述した実施例と同様の作用効果が得られる。

【0059】(b)第2実施例の説明図8は本発明の第2実施例としてのベクトル処理装置を示すブロック図で、この図8において、21はインターリーブされた複数のバンク単位に複数のエレメント・データを記憶するベクトル・レジスタ（VR）、22-1, 22-2, ..., 22-nはそれぞれベクトル・レジスタ21上のデータを入力オペランドとするかもしくは演算結果をベクトル・レジスタ21に書き込む演算パイプラインである。

【0060】また、23-1, 23-2, ..., 23-mはベクトル・レジスタ21と主記憶部24との間において各エレメント・データを高速にロードないしストアすべくパイプライン構成されたメモリ・アクセス・パイプラインで、各メモリ・アクセス・パイプライン23-1, 23-2, ..., 23-mは、ロード・パイプライン23Aおよびストア・パイプライン23Bとしての機能を有している。ここで、ロード・パイプライン23Aは、主記憶部24からベクトル・レジスタ21へデータを転送（ロード）するためのものであり、ストア・パイプライン23Bは、ベクトル・レジスタ21に格納されたデータを主記憶部24へ転送（ストア）するためのものである。

【0061】なお、図8で示した各メモリ・アクセス・パイプライン23-1, 23-2, ..., 23-mでは、ロード・パイプライン23Aとストア・パイプライン23Bとがペアとなってそなえられているが、いずれか一方のみを有する構成としてもよい。また、メモリ・アクセス・パイプライン23-1, 23-2, ..., 23-mのいずれかがロード・パイプライン23Aもしくはストア・パイプライン23Bの一方のみを有する構成としてもよい。

【0062】25は追越し防止制御部で、この追越し防止制御部25は、メモリ・アクセス・パイプライン23-1, 23-2, ..., 23-mにおけるロード・パイプライン23Aからベクトル・レジスタ21へデータを転送する命令の実行中に、ロード・パイプライン23Aがベクトル・レジスタ21に書き込んだデータを入力オペランドとする後続の演算命令を、演算パイプライン22

−1, 22−2, …, 22−nのいずれかが実行する場合、命令の実行順序を保証するために、ロード・パイプライン23Aの実行を後続の演算パイプラインの処理が追い越す条件を検出した時に、全ての演算パイプライン22−1, 22−2, …, 22−nの実行を一時中断すべく、例えば、処理の一時中断を要求している間は“1”に立ち上がり、処理を続行可能な時には“0”になる追越し防止制御信号を出力するものである。

【0063】27は各演算パイプライン22−1, 22−2, …, 22−nや各メモリ・アクセス・パイプライン23−1, 23−2, …, 23−mに対する命令の発信と進行状況とを管理する命令発信/管理部で、この命令発信/管理部27は、命令を発信するときに、その命令の発信以前に実行を開始され未だその実行を完了していない命令との間にオペランド・レジスタのリンクがあるかどうかをチェックし、リンクがあれば、追越し防止制御部25にリンク条件が成立したことをリンク情報として通知する機能を有している。

【0064】そして、第2実施例では、図9に示すような追越し防止制御信号変更部26Aおよび収束期間通知部26Bが、各演算パイプライン22−1, 22−2, …, 22−n毎にそなえられている。この追越し防止制御信号変更部26Aは、ベクトル・レジスタ21からのデータ供給を受けるリード処理期間と、リード処理期間後に結果をまとめ上げる収束期間とを必要とするベクトル命令については、該当ベクトル命令の収束処理を実行中の演算パイプライン22−1, 22−2, …, 22−nに対する追越し防止制御を行なわないように、追越し防止制御部25から出力される追越し防止制御信号を変更するものである。

【0065】また、収束期間通知部26Bは、各演算パイプライン22−1, 22−2, …, 22−nが有する収束期間の開始信号(start final sequence;例えば収束期間開始時に立ち上がるパルス信号)と収束期間の終了信号(terminal final sequence;例えば収束期間終了時に立ち上がるパルス信号)とに基づいて、各演算パイプライン22−1, 22−2, …, 22−nが前述した収束期間にあるか否かを、収束期間通知信号として対応する追越し防止制御信号変更部26Aへ出力するもので、例えば、図9に示すように構成されている。

【0066】図9において、26aはORゲート、26bはANDゲート、26cはフリップフロップ、26eはインバータ(NOTゲート)であり、ORゲート26aは、開始信号とフリップフロップ26cからのデータ出力との論理和をとるものであり、ANDゲート26bは、ORゲート26aからの論理和出力と、終了信号のインバータ26eによる反転信号との論理積をとるものである。

【0067】また、フリップフロップ26cは、ANDゲート26bからの論理積出力に応じてデータ出力を

“1”に立ち上げるもので、ORゲート26aに開始信号が入力されると、フリップフロップ26cのデータ出力は立ち上がって“1”にセットされる一方、ANDゲート26bに終了信号が入力されると、フリップフロップ26cのデータ出力は“1”から“0”にリセットされるようになっている。つまり、フリップフロップ26cのデータ出力(収束期間通知手段)は、各演算パイプライン22−1, 22−2, …, 22−nが収束処理を実行中(収束期間中)、“1”に立ち上がっている。

【0068】そして、追越し防止制御信号変更部26AをなすANDゲート26dは、追越し防止制御部25により生成される追越し防止制御信号と、フリップフロップ26cのデータ出力(収束期間通知手段)のインバータ(NOTゲート)26fによる反転信号との論理積をとるもので、フリップフロップ26cのデータ出力が“1”の間は、収束期間であるので、追越し防止制御信号は、ANDゲート26dで強制的に“0”に変更される一方、フリップフロップ26cのデータ出力が“0”の時は、追越し防止制御信号はそのまま出力されるようになっている。

【0069】次に、上述のごとく構成されているベクトル処理装置の動作について、図10～図18を参照しながら説明する。

①〔該当命令の動作の説明〕

まず、該当命令の動作に関して説明する。ただし、実際のインプリメンテーションの都合により、ここに記す動作とは若干異なる動作をすることもある。ベクトル・レジスタ21上のデータの総和を求める総和演算命令や、ベクトル・レジスタ21上のデータの最大値もしくは最小値を求める検索演算命令では、元となるデータをベクトル・レジスタ21から供給を受ける。

【0070】総和演算命令では、順次、累和を求めることになるが、処理速度の向上のために、パイプラインのステージ分の部分累和を、多重にもつ演算器分だけ生成する。次に、総和演算の結果を求めるために、これらの部分積を全て足し込む。総和演算では、この部分積を足し込む処理を実行する期間が収束期間となる。検索演算命令では、順次、比較し選択することになるが、この場合も、ステージ数分の部分的な選択結果が、多重にもつ演算器分だけ生成された後、これらの部分的な選択結果の中から最終的に1つの結果を求める。検索演算では、この部分的な選択結果の中から最終的な1つの結果を求める操作を実行する期間が収束期間となる。

【0071】これらの収束期間における処理は、ベクトル・レジスタ21からのデータ供給を受けない処理のため、ベクトル・レジスタ21のリンクによる追越し防止制御は不要である。

②〔一般的な追越し防止制御の説明〕

追越し防止制御を司るベクトル処理装置の部分は、一例をあげると次のようになっている。命令の発信と進行状

況とを管理する命令発信／管理部27が命令を発信するときに、その命令の発信以前に実行を開始され、未だその実行を完了していない命令との間にオペランド・レジスタのリンクがあるかどうかをチェックし、リンクがあれば、追越し防止制御部25にリンク条件が成立したことを通知する。

【0072】このリンク条件は、リンクの成立したベクトル・レジスタ21にデータを書き込む先行命令の実行を完了して、レジスタ・リンクが解消したときに解除される。レジスタ・リンクが複数存在するときは、全てのレジスタ・リンクが解消されたときにリンク条件を解除する。リンク条件の成立中は、先行命令のデータ書込が中断されることを検出すると、追越し防止制御部25が起動される。この追越し防止制御部25は、全ての演算パイプライン22-1, 22-2, ..., 22-nとストア・パイプライン23Bとに対して処理を一時中断させるように通知する。ただし、リンクの起点（親）となるパイプラインについては、処理の中断は通知しない。先行命令のデータ書込を再開するか、もしくは、先行命令を完了すると、追越し防止制御部25は、全ての演算パイプライン22-1, 22-2, ..., 22-nとストア・パイプライン23Bとに対して処理を再開するように通知する。

【0073】実際には、各演算パイプライン22-1, 22-2, ..., 22-nのベクトル・レジスタ21からのデータ読出のスループットは、どの演算パイプライン22-1, 22-2, ..., 22-nでも等しいようにして、処理スループットの差による動的な追越し防止制御を不要としている。このため、スループットの変化するリンクの親となるパイプラインはロード・パイプライン23Bに限定される。演算パイプライン22-1, 22-2, ..., 22-n間の追越し防止については、全ての演算パイプライン22-1, 22-2, ..., 22-nに対して同時に処理の中断と再開とを指示することで回避されている。

【0074】なお、インプリメンテーションの都合により、実際には、低スループットの演算パイプラインもベクトル処理装置上に存在している。このような場合には、通常は、この低スループットのパイプラインは、他のパイプラインとのリンク動作を禁止するように命令発信の段階で制御されている。とは言っても、より性能を追求する制御では、低スループットのパイプラインと言えども、制限付きでリンク動作をさせることがある。

【0075】一例を挙げれば、先行する命令を処理するパイプラインのスループットが高いか等しければ、そのパイプラインの書込結果を読み出すような形のリンク動作を行なうことは可能である。禁止されているのは、低スループットの演算パイプラインの書込結果を使用する形で、高スループットのパイプラインが該低スループットのパイプラインにリンクして動作することである。

【0076】③〔リード処理期間の説明〕

リード処理期間は、命令発信／管理部27からの命令で最初に処理するデータを演算パイプライン22-1, 22-2, ..., 22-nがベクトル・レジスタ21から受け取った時点を開始とし、その命令で最後に処理するデータを演算パイプライン22-1, 22-2, ..., 22-nがベクトル・レジスタ21から受け取った時点を終了とする。

【0077】④〔追越し防止制御信号変更部26A（ANDゲート26d）を追越し防止制御部25側に、収束期間通知部26Bを命令発信／管理部27側にそなえた場合の説明〕

ここでは、命令発信／管理部27から追越し防止制御部25に対して、各演算パイプライン22-1, 22-2, ..., 22-nの処理が収束期間に入ったことを演算パイプライン情報とともに通知する機能と、収束期間が完了したことを演算パイプライン情報とともに通知する機能が追加されている。つまり、図9により前述した収束期間通知部26Bとしての機能が命令発信／管理部27にそなえられている。

【0078】また、追越し防止制御部25には、各演算パイプライン22-1, 22-2, ..., 22-nに対する処理の一時中断（追越し防止制御信号）を通知する機能が各演算パイプライン22-1, 22-2, ..., 22-n毎に設けられているが、その追越し防止制御部25における各追越し防止制御信号出力用信号線毎に、追越し防止制御信号変更部26Aが設けられ、演算パイプライン22-1, 22-2, ..., 22-nのうち、収束期間にある演算パイプラインには処理の一時中断（追越し防止制御信号）が通知されないようになっている。

【0079】具体的には、命令発信／管理部27では、命令をデコードして収束期間をもつ命令であることを検出すると、その命令を発信する演算パイプライン22-1, 22-2, ..., 22-nの管理部（図示せず）に収束期間を有する命令である旨のフラグを立てる。演算パイプライン側管理部では、ベクトル長からリード期間に要する時間を計算し、追越し防止制御部25からの処理の一時中断を通知する機能による情報を使用して、リード処理期間を正確に把握する。

【0080】リード処理期間の完了とともに、収束期間が開始されるわけであるから、この時に追越し防止制御部25に対して、収束期間通知部26Bにより、その演算パイプライン22-1, 22-2, ..., 22-nが収束期間に入ったことを通知する。このとき通知される収束期間通知信号は、前述した通り、収束期間にあるときにはその演算パイプライン22-1, 22-2, ..., 22-nに対応する信号線が“1”に立ち上がり、それ以外のときは“0”となるものである。

【0081】そして、追越し防止制御信号出力用の信号線上に、前述のように追越し防止制御信号変更部26A

をなすANDゲート26dを設けることで、対応する演算パイプライン22-1, 22-2, ..., 22-nからの収束期間通知信号(フリップフロップ26cのデータ出力)が“0”のときには、追越し防止制御信号は、そのままANDゲート26dを通過して対応する演算パイプライン22-1, 22-2, ..., 22-nに通知される一方、対応する演算パイプライン22-1, 22-2, ..., 22-nからの収束期間通知信号が“1”のときには、追越し防止制御信号は、ANDゲート26dにより“0”に変更されてから対応する演算パイプライン22-1, 22-2, ..., 22-nに通知される。

【0082】これにより、従来、図10に示すように、追越し防止制御信号の出力時には収束期間であっても完全に中断されていた処理が、本実施例では、収束期間中にある演算パイプラインについては変更された追越し防止制御信号を受けることにより、図11に示すように、中断されことなく実行されるようになる。

⑥〔追越し防止制御信号変更部26Aおよび収束期間通知部26Bを各演算パイプライン側にそなえた場合の動作の説明〕

ここでは、各演算パイプライン22-1, 22-2, ..., 22-n毎に追越し防止制御信号変更部26Aおよび収束期間通知部26Bをそなえ、各演算パイプライン自身のシーケンサ(開始信号、終了信号)に基づいて、収束期間通知部26Bにより自分が収束期間にいることを検出し、収束期間中は、追越し防止制御信号変更部26Aにより、追越し防止制御部25から各演算パイプライン22-1, 22-2, ..., 22-nに対する追越し防止制御信号を、“1”から“0”に変更するようにしている。

【0083】なお、各演算パイプライン22-1, 22-2, ..., 22-nには、命令、ベクトル長、起動信号を受け取って処理を実行できるようにデータフローを制御するための内部シーケンサ(図示せず)が用意されている。この内部シーケンサは、データフローを制御するための命令の各種シーケンスを実行できるように、命令に伴うシーケンス連鎖の手順を滞りなく実行できるようになっている。

【0084】収束期間通知部26Bでは、リード処理期間の終了と収束期間開始との情報を内部シーケンサから受け取って、収束期間以外のときは、追越し防止制御信号変更部26A(ANDゲート26d)により、追越し防止制御信号をそのまま通過させるが、対応する演算パイプラインに対する収束期間のときには、追越し防止制御信号を“0”に変更する。

【0085】これにより、上述のように追越し防止制御信号変更部26Aおよび収束期間通知部26Bを各演算パイプライン側にそなえた場合にも、従来、図10に示すように、追越し防止制御信号の出力時には収束期間であっても完全に中断されていた処理が、本実施例では、

収束期間中にある演算パイプラインについては変更された追越し防止制御信号を受けることにより、図11に示すように、中断されことなく実行されるようになる。

【0086】⑥〔演算パイプラインの一構成例およびその動作の説明〕

ここで、演算パイプラインの一例として、図13に示すように4段のステージから成るパイプライン式の加算器22a~22dを、図12に示すように4個有する総和演算パイプライン22Aを示す。なお、図13において、28aは第1ステージ・レジスタ、28bは第2ステージ・レジスタ、28cは第3ステージ・レジスタ、28dは第4ステージ・レジスタ、28eは転送用中継レジスタ、29はセクタ、30は指数差計算手段、31は桁合わせ手段、32は加算/減算手段、33は正規化手段である。また、各加算器22a~22dのステージ段数や、加算器の数については、処理装置毎に最適な段数が用意され、処理する命令(検索演算命令等)によっては加算器を比較選択手段とすることもできる。

【0087】各加算器22a~22dの第1ステージは桁合わせに先立つ指数比較処理、第2ステージは桁合わせ処理を実行する。第3ステージは加算/減算処理を実行する。第4ステージは正規化処理である。加算器22aはベクトル・レジスタ中のデータ・エレメント番号の4による剰余が0のエレメントを処理し、加算器22bはエレメント番号の4による剰余が1のエレメントを処理し、加算器22cはエレメント番号の4による剰余が2のエレメントを処理し、加算器22dはエレメント番号の4による剰余が3のエレメントを処理するように構成されている。

【0088】また、各加算器22a~22dを一つにまとめるために、加算器22b~22dから加算器22aに対して結果を転送することができるようになっている。総和演算では、ベクトル・レジスタ21からのデータ供給を受けるリード処理期間中は、第4ステージのデータは第1ステージに戻されて、加算器22a~22dは累和加算器として動作する。また、4つの加算器22a~22dは並列に動作する。最終リード処理期間中は各加算器22a~22d中に4つの部分和を生成する。

【0089】加算器22aには、 $\Sigma(E_{111})$, $\Sigma(E_{1111})$, $\Sigma(E_{11111})$, $\Sigma(E_{111111})$ の4種類の部分和が生成される。ここで、 E_i はエレメント番号iのデータ・エレメントの値である。収束期間では、各部分和を取りまとめる手段は、幾つも存在しえるが、ここでは、その中の一例を図14に示す。

【0090】まず、各加算器22a~22d中で、4つの部分和を足し合わせて各々1つの部分結果を生成する。次に、これらの4つの部分結果を足し合わせて最終的な結果を演算する。ここでも、加算器22aを例にとって、部分結果を生成する様子を説明する。収束期間に入った瞬間に、どの部分和がどのステージにいるかにつ

いては、VL長に依存するので、 $\Sigma(E_{iii})$ の部分
和が第1ステージ・レジスタ28aにセットされるまで空
足しを行ない演算順序を保証する。第1ステージ・レ
ジスタ28aで2 τ の間、部分和 $\Sigma(E_{iii})$ をホールド
する。

【0091】その間に、第1ステージのもう一方のレ
ジスタ28aに部分和 $\Sigma(E_{iiii})$ をセットする。部分
和 $\Sigma(E_{iii})$ と部分和 $\Sigma(E_{iiii})$ を第1ステー
ジから第2ステージに移動させ加算処理を開始するととも
に、部分和 $\Sigma(E_{iiii})$ を第1ステージ・レジスタ2
8aにセットし、2 τ の間、ホールドする。部分和 Σ
(E_{iiii})をホールド中に、もう一つの第1ステージ
・レジスタ28aに部分和 $\Sigma(E_{iiii})$ をセットす
る。そして、部分和 $\Sigma(E_{iiii})$ と部分和 $\Sigma(E$
 $_{iiii})$ とを加算する。

【0092】続いて、部分和 $\{\Sigma(E_{iii}) + \Sigma(E$
 $_{iiii})\}$ を第1ステージ・レジスタ28aにセッ
トし、3 τ の間、ホールドする。その間に部分和 $\{\Sigma(E$
 $_{iiii}) + \Sigma(E_{iiii})\}$ をもう一方の第1ステー
ジ・レジスタ28aにセットして加算処理を実行して加算
器22aの中間結果を求める。収束期間における、この
中間和を求める動作は、各加算器22a~22dで並行
して行なわれる。

【0093】最終的な結果を求めるための演算は、加算
器22aで実行する。加算器22aでは、第1ステー
ジ・レジスタ28aに加算器22aの中間和をホールド
し、加算器22bの中間和がもう一方の第1ステー
ジ・レジスタ28aにセットされるのを待つ。そして、加算
器22aと加算器22bとの中間和どうしを加算して中
間和Aを求める。この中間和Aは、第1ステージ・レ
ジスタ28a上でホールドされ、加算器22cの中間和が
もう一方の第1ステージ・レジスタ28aにセットされ
るのを待つ。そして、中間和Aと加算器22cの中間和
が加算されて、中間和Bが生成される。この中間和B
は、第1ステージ・レジスタ28aでホールドされ、加
算器22dの中間和がもう一方の第1ステージ・レ
ジスタ28aにセットされるのを待つ。そして、中間和Bと
加算器22dの中間和とを加算して、最終的な結果を生
成する。

【0094】⑦〔付加演算器付き演算パイプラインに本
発明を適用した実施例の説明〕

次に、各演算パイプライン22-1, 22-2, ..., 22-nが、基本演算器と収束を処理する付加演算器とを有する構成のもので(例えば図17参照)、収束処理を付加演算器により実行し、収束処理中、基本演算器により後続の他の演算命令を実行できるものである場合について説明する。

【0095】このような場合も、基本演算器から切り離されて収束期間に入ってから、その命令を完了するまでの収束期間中、付加演算器では、追越し防止制御部25

から各演算パイプライン22-1, 22-2, ..., 22-nに対する処理の一時中断を通知する手段による情報(追越し防止制御信号)を、追越し防止制御信号変更部26Aにより受け付けないようにしている。

【0096】付加演算器を使用する場合には、内部シーケンサについても、基本演算器部分と付加演算器部分とに分割されている。付加演算器部分の内部シーケンサは、基本演算器部分の内部シーケンサから起動され、収束期間に入る時点で基本演算器部分から切り離される旨の通知を受ける。付加演算器部分の内部シーケンサは、基本演算器部分から切り離されると収束のためのシーケンスを起動する。

【0097】収束期間通知部26Bは、リード期間の終了と収束期間開始との情報を付加演算器部分の内部シーケンサから受け取って、追越し防止制御信号変更部26Aにより、収束期間以外のときは、追越し防止制御信号をそのまま通過させるが、対応する演算パイプライン22-1, 22-2, ..., 22-nの付加演算器に対する収束期間の時には追越し防止制御信号を“0”に変更する。

【0098】基本演算器部分については、付加演算器部分を切り離した後は、付加演算器部分が演算を完了するまで、付加演算器部分を必要としない演算を実行可能として、命令発信/管理部27からの起動待ちとなる。付加演算器部分が命令を完了し且つ基本演算器部分が付加演算器部分を使用しない命令のリード処理期間を完了するか、付加演算器部分と基本演算器部分との双方ともに命令の実行を完了していれば、その演算パイプライン22-1, 22-2, ..., 22-nは、付加演算器を使用する命令を実行可能になる。

【0099】これにより、収束処理を行なう付加演算器付きの演算パイプラインについても、従来、図15に示すように、追越し防止制御信号の出力時には収束期間であっても完全に中断されていた処理が、本実施例では、収束期間中にある演算パイプラインの付加演算器については変更された追越し防止制御信号を受けることにより、図16に示すように、中断されることなく実行されるようになる。

【0100】⑧〔付加演算器付き演算パイプラインの一構成例およびその動作の説明〕

ここでは、付加演算器付き演算パイプラインの一例として、図17に示すように、4段のステージから成るパイプライン式の基本演算器(図13に示した加算器と同様構成のもの)34a~34dと、4段のステージから成るパイプライン式の付加演算器(図13に示した加算器と同様構成のもの)35a~35dとを組み合わせた複合演算器36a~36dを4つもつような総和演算パイプライン22Bを示す。

【0101】なお、ステージ段数や加算器の数については、処理装置毎に最適な段数が用意され、処理する命令

(検索演算命令等) によっては加算器を比較選択手段とすることもできる。また、前述したものと同様に、各演算器(加算器) 34a~34dおよび35a~35dの第1ステージは桁合わせに先立つ指数比較処理、第2ステージは桁合わせ処理を実行する。第3ステージは加算/減算処理を実行する。第4ステージは正規化処理である。

【0102】複合演算器36aはベクトル・レジスタ21中のデータ・エレメント番号の8による剰余が0と1のエレメントを処理し、複合演算器36bはエレメント番号の8による剰余が2と3のエレメントを処理し、複合演算器36cはエレメント番号の8による剰余が4と5のエレメントを処理し、複合演算器36dはエレメント番号の8による剰余が6と7のエレメントを処理するように構成されている。

【0103】また、各付加加算器35a~35dの結果を一つにまとめるために、付加加算器35b、付加加算器35c、付加加算器35dから付加加算器35aに対して結果を転送できるようになっている。そして、リード処理期間には、基本演算器34a~34dからなる基本演算部34では、連続する2つのエレメントの和を求めて、付加演算器35a~35dからなる付加演算部35に転送する。付加演算部35は基本演算部34から受け取った中間和を足し込む。リード期間が終了するときには、付加演算部内に4つの部分和が生成されている。複合演算器36aを例にとると、この4つの部分和は、 $\Sigma(E_{321} + E_{3211})$, $\Sigma(E_{32115} + E_{32119})$, $\Sigma(E_{32116} + E_{32117})$, $\Sigma(E_{32124} + E_{32125})$ となる。ここで、 E_i はエレメント番号iのデータ・エレメントの値である。

【0104】収束期間に入ると、付加演算部35は基本演算部34のデータバスからのデータを受け付けないようにする。続いて、4つの部分和から中間結果を計算し、最後に4つの中間結果から最終結果を計算する。複合演算器36aを例にとって中間結果を求める手順を図18により説明する。

【0105】まず、演算順序を保証するために、部分和 $\Sigma(E_{321} + E_{3211})$ が第1ステージ・レジスタ28a上に来るまで空足しを行なう。その部分和は第1ステージ・レジスタ28a上で2τ間ホールドされる。その間に部分和 $\Sigma(E_{32115} + E_{32119})$ をもう一つの第1ステージ・レジスタ28aにセットする。そして、部分

和Aの計算を開始する。
【0106】次に、部分和 $\Sigma(E_{32116} + E_{32117})$ を第1ステージ・レジスタ28aにセットし2τの間ホールドする。その間に部分和 $\Sigma(E_{32124} + E_{32125})$ をもう一方の第1ステージ・レジスタ28aにセットする。そして、もう一つの部分

和スタ28aにセットし、そして中間結果を計算する。

【0107】収束期間における、この中間結果を求める動作は、各付加演算器35a~35dで並行して行なわれる。最終的な結果を求めるための演算は、付加演算器35aで実行する。付加演算器35aでは、第1ステージ・レジスタ28aに付加演算器35aの中間結果をホールドし、付加演算器35bの中間結果がもう一方の第1ステージ・レジスタ28aにセットされるのを待つ。そして、付加演算器35aと付加演算器35bの中間結果どうしを加算して中間結果Aを求める。この中間結果Aは、第1ステージ・レジスタ28a上でホールドされ、付加演算器35cの中間結果がもう一方の第1ステージ・レジスタ28aにセットされるのを待つ。

【0108】そして、中間結果Aと付加演算器35cの中間結果とを加算されて中間結果Bが生成される。この中間結果Bは、第1ステージ・レジスタ28aでホールドされ、付加演算器35dの中間結果がもう一方の第1ステージ・レジスタ28aにセットされるのを待つ。そして、中間結果Bと付加演算器35dの中間結果とを加算して、最終的な結果を生成する。

【0109】このように、本発明の第2実施例のベクトル処理装置によれば、追越し防止制御部25から追越し防止制御信号が出力された際に、演算パイプライン22-1, 22-2, ..., 22-n, 22A, 22Bが収束処理のシーケンスを実行中で、収束期間条件が成立している間は、追越し防止制御信号変更部26Aにより追越し防止制御部25からの追越し防止制御信号が変更され、収束処理中の演算パイプライン22に対する追越し防止制御が禁止されるので、追越し防止制御信号変更部26Aおよび収束期間通知部26Bという極めて少量の物量の増加によるだけで、収束期間におけるレジスタ・リンクによる追越し制御のオーバーヘッドを回避することができ、性能の向上を実現することができる。

【0110】

【発明の効果】以上詳述したように、本発明のベクトル処理装置(請求項1, 2)によれば、スループットの少ない演算パイプラインを、1つのバンクスロットしか使用しないメモリ・アクセス・パイプライン(ストア・パイプライン)と共用し、演算パイプラインをオーバーラップさせて実行することにより、演算スループットの大幅な向上を実現できる効果がある。

【0111】また、本発明のベクトル処理装置(請求項3~5)によれば、追越し防止制御部25から追越し防止制御信号が出力された際に、演算パイプラインが収束処理のシーケンスを実行中で、収束期間条件が成立している間は、変更部追越し防止制御信号が変更され、収束処理中の演算パイプラインに対する追越し防止制御が禁止されるので、処理速度の大幅な改善を実現できる効果がある。

【図面の簡単な説明】

【図 1】第 1 の発明の原理説明図である。

【図 2】第 2 の発明の原理ブロック図である。

【図 3】本発明の第 1 実施例としてのベクトル処理装置を示すブロック図である。

【図 4】第 1 実施例の動作を説明するためのタイミングチャートである。

【図 5】第 1 実施例のバンク管理部の構成例を示すブロック図である。

【図 6】第 1 実施例の動作を説明するためのタイミングチャートである。

【図 7】第 1 実施例の動作を説明するためのタイミングチャートである。

【図 8】本発明の第 2 実施例としてのベクトル処理装置を示すブロック図である。

【図 9】第 2 実施例における追越し防止制御信号変更部の構成を示す回路図である。

【図 10】第 2 実施例の動作を説明するための図である。

【図 11】第 2 実施例の動作を説明するための図である。

【図 12】第 2 実施例の演算パイプラインの構成例を示すブロック図である。

【図 13】第 2 実施例の加算器の構成例を示すブロック図である。

【図 14】第 2 実施例の演算パイプラインの動作例を説明するためのタイミングチャートである。

【図 15】第 2 実施例の動作を説明するための図である。

【図 16】第 2 実施例の動作を説明するための図である。

【図 17】第 2 実施例の付加演算器付き演算パイプラインの構成例を示すブロック図である。

【図 18】第 2 実施例の付加演算器付き演算パイプラインの動作例を説明するためのタイミングチャートである。

【図 19】一般的なバンクスロットのタイミング設定例を示すタイミングチャートである。

【符号の説明】

1-0, 1-1, ..., 1-n ベクトル・レジスタ
2 メモリ・アクセス・パイプライン
2A ロード・パイプライン
2B ストア・パイプライン
3A, 3B-0, 3B-1, ..., 3B-m 書込レジスタ
4A, 4B-0, 4C-0, 4B-1, 4C-1, ...,
4B-m, 4C-m 読出レジスタ

* 5-0, 5-1, ..., 5-m 演算パイプライン

6 命令制御部

7 バンク管理部

7a バンクスロット・カウンタ

11-1, 11-2, 11-3 管理レジスタ

12 バンクスロット割当回路

13 通知レジスタ

14 起動信号制御部

15 割算パイプライン

10 15a~15f 割算器

20 主記憶部

21 ベクトル・レジスタ

22, 22-1, 22-2, ..., 22-n 演算パイプライン

22A, 22B 総和演算パイプライン

22a~22d 加算器

23-1, 23-2, ..., 23-m メモリ・アクセス・パイプライン

23, 23A ロード・パイプライン

20 23B ストア・パイプライン

24 主記憶部

25 追越し防止制御部

26 変更部

26A 追越し防止制御信号変更部

26B 収束期間通知部

26a ORゲート

26b ANDゲート

26c フリップフロップ

26d ANDゲート

30 26e, 26f インバータ (NOTゲート)

27 命令発信/管理部

28a 第 1 ステージ・レジスタ

28b 第 2 ステージ・レジスタ

28c 第 3 ステージ・レジスタ

28d 第 4 ステージ・レジスタ

28e 転送用中継レジスタ

29 セレクタ

30 指数差計算手段

31 桁合わせ手段

40 32 加算/減算手段

33 正規化手段

34 基本演算部

34a~34d 基本演算器

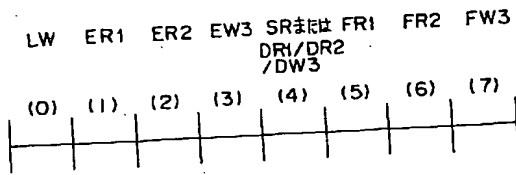
35 付加演算部

35a~35d 付加演算器

* 36a~36d 複合演算器

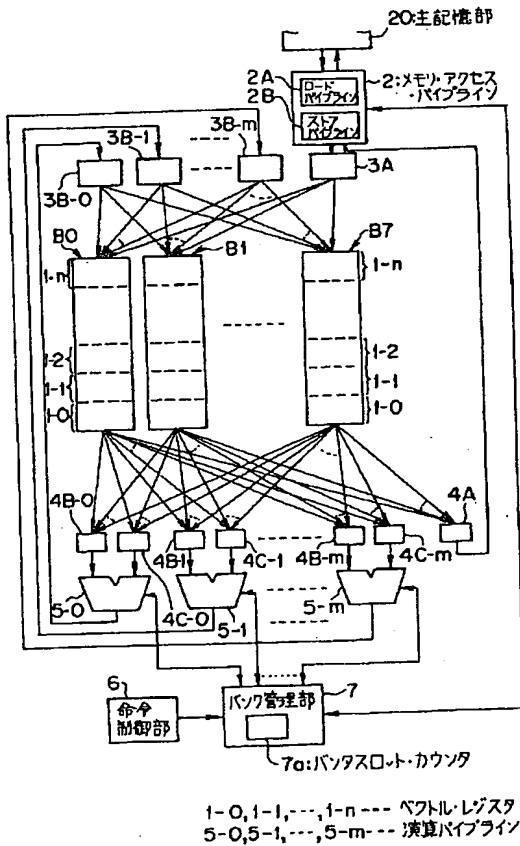
【図1】

第1の発明の原理説明図



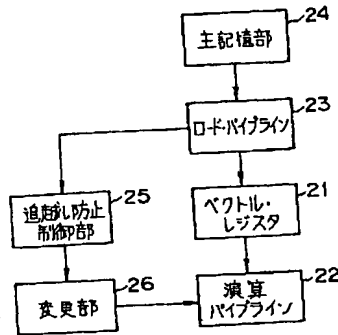
【図3】

本発明の第1実施例としてのベクトル処理装置を示すブロック図



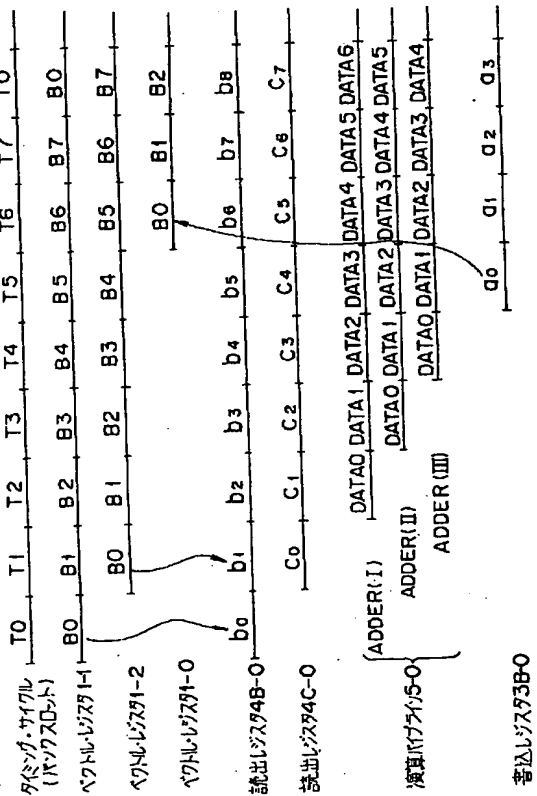
【図2】

第2の発明の原理ブロック図



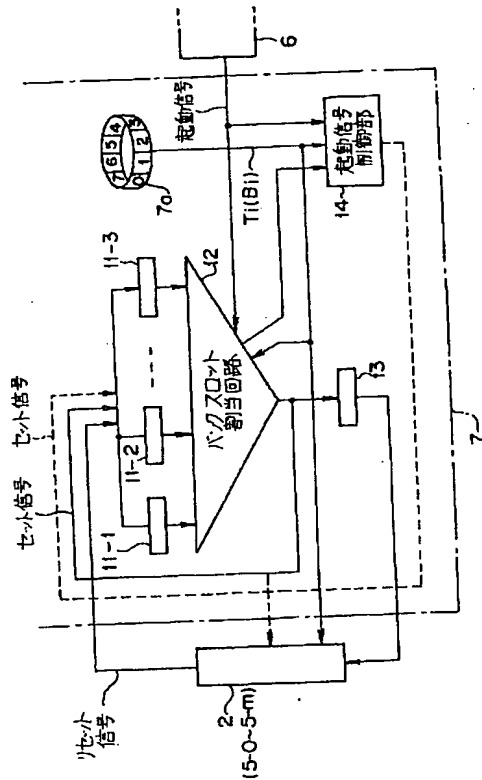
【図4】

第1実施例の動作を説明するためのタイミングチャート



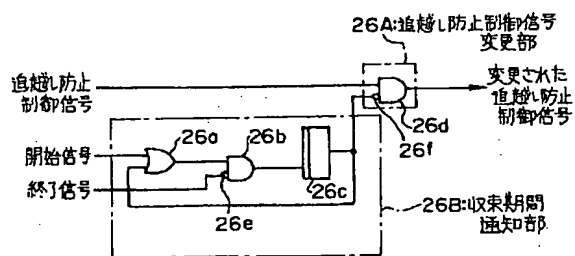
【図 5】

第1実施例のバンク管理部の構成例を示すブロック図



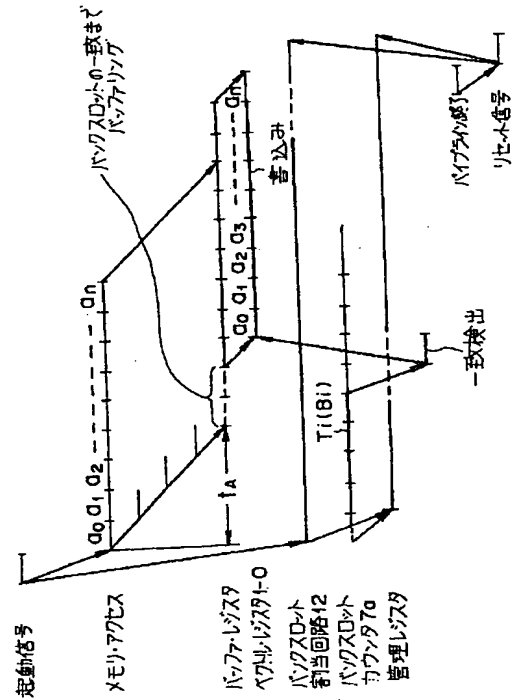
【図 9】

第2実施例における追越し防止制御信号変更部の構成を示す回路図



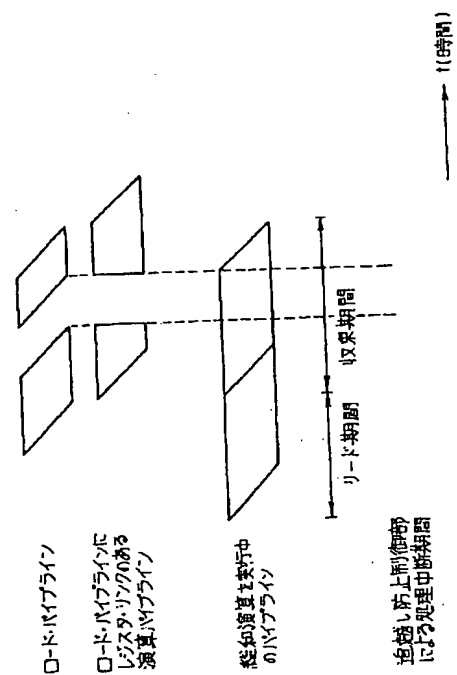
【図 6】

第1実施例の動作を説明するためのタイミングチャート

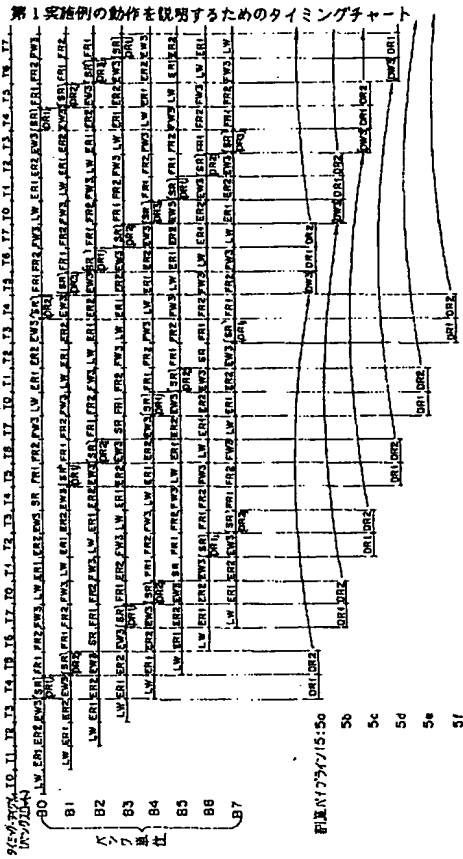


【図 1 1】

第2実施例の動作を説明するための図

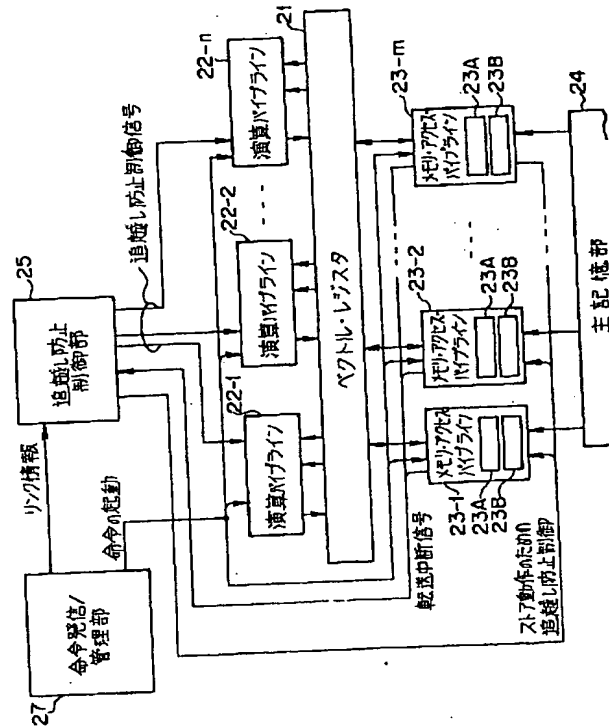


【図7】



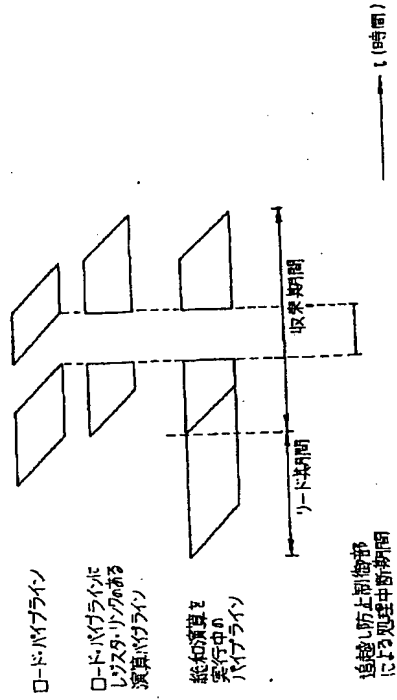
【図8】

本発明の第2実施例としてのベクトル処理装置を示すブロック図



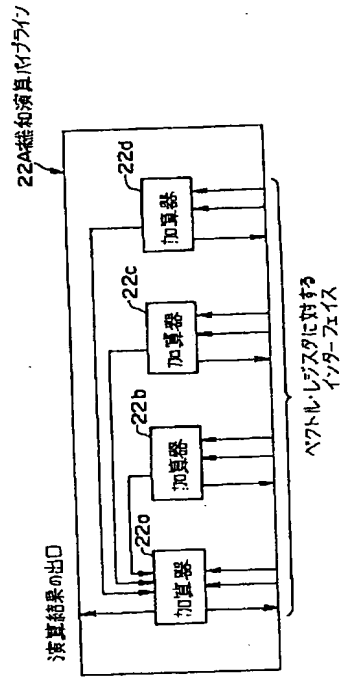
【図10】

第2実施例の動作を説明するための図

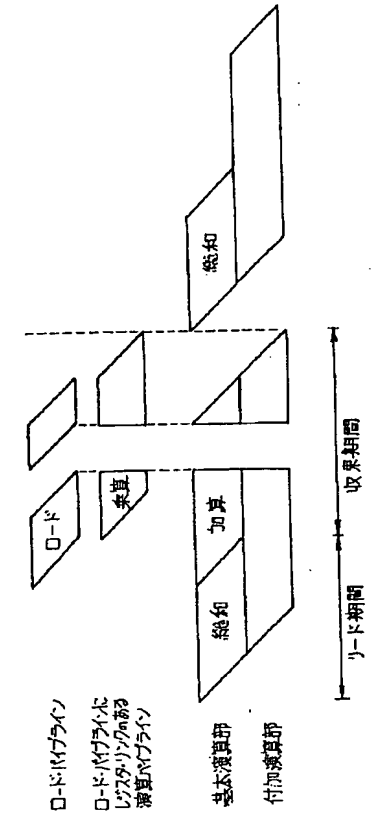


【図12】

第2実施例の演算パイプラインの構成例を示すブロック図

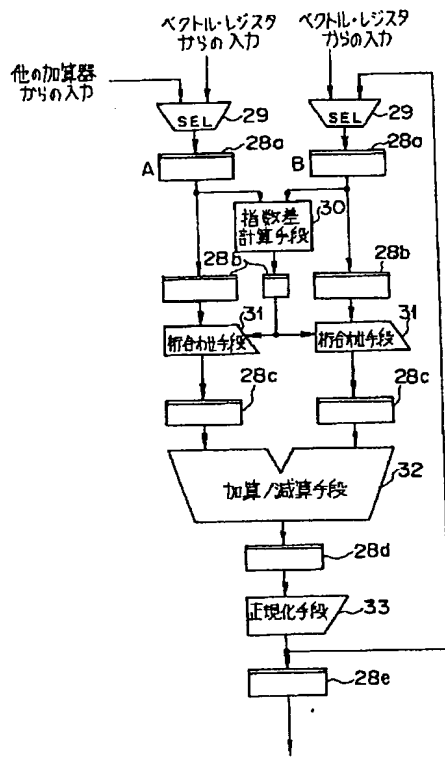


【図15】



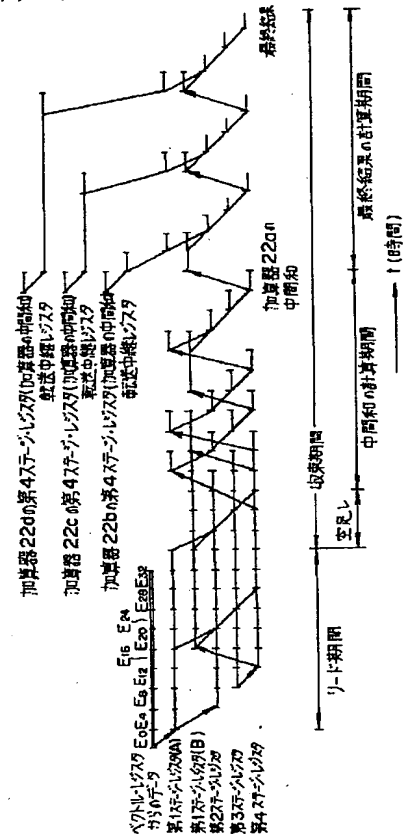
【図 13】

第2実施例の加算器の構成例を示すブロック図



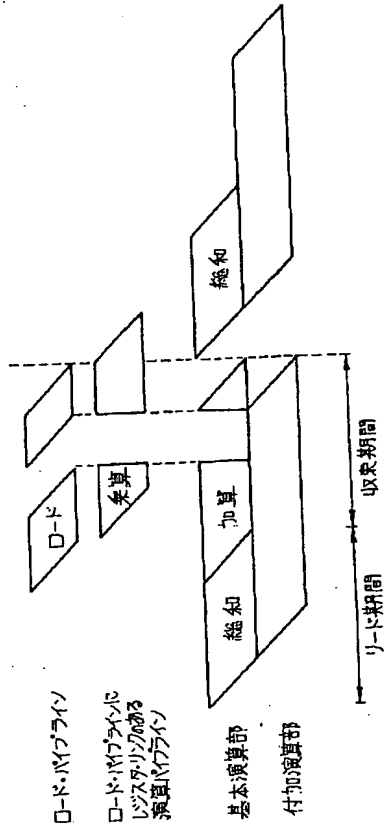
【図 14】

第2実施例の演算パイプラインの動作例を説明するためのタイミングチャート

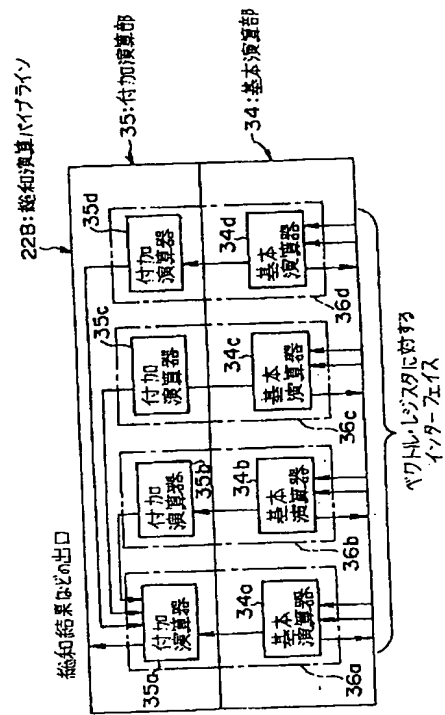


【図 16】

第2実施例の動作を説明するための図

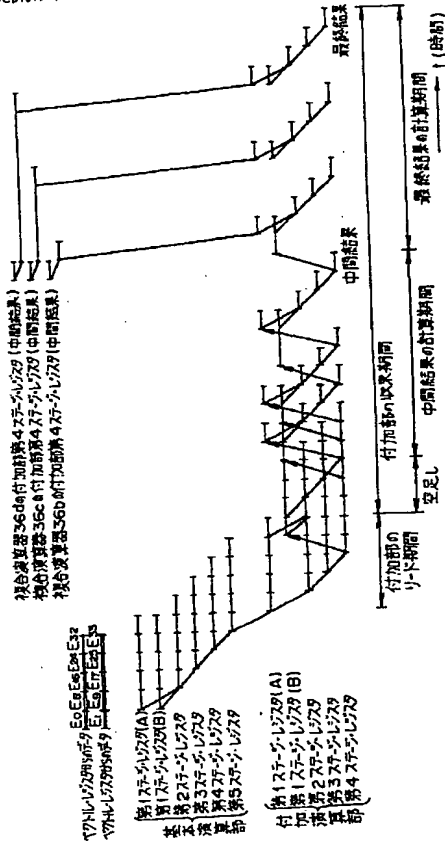


【図 17】

第2実施例の付加演算器付き演算パイプラインの
構成例を示すブロック図

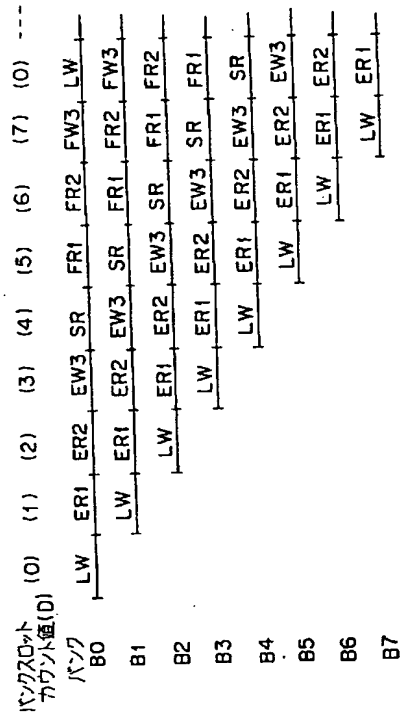
【図 18】

第2実施例の付加演算器付き演算パイプラインの動作例を説明する
ためのタイミングチャート



【図 19】

一般的なバックスロットのタイミング設定例を示す
タイミングチャート



フロントページの続き

(72)発明者 今野 勝彦

神奈川県川崎市中原区上小田中1015番地
富士通株式会社内

(72)発明者 渥美 宏昭

神奈川県川崎市中原区上小田中1015番地
富士通株式会社内